

MMM	MMM	TTTTTTTTTTTTTT	AAAAAAAAA	AAAAAAAAA	CCCCCCCCCCCC	PPPPPPPPPPPP	
MMM	MMM	TTTTTTTTTTTTTT	AAAAAAAAA	AAAAAAAAA	CCCCCCCCCCCC	PPPPPPPPPPPP	
MMM	MMM	TTTTTTTTTTTTTT	AAAAAAAAA	AAAAAAAAA	CCCCCCCCCCCC	PPPPPPPPPPPP	
MMMMMM	MMMMMM	TTT	AAA	AAA	CCC	PPP	PPP
MMMMMM	MMMMMM	TTT	AAA	AAA	CCC	PPP	PPP
MMMMMM	MMMMMM	TTT	AAA	AAA	CCC	PPP	PPP
MMM	MMM	TTT	AAA	AAA	CCC	PPP	PPP
MMM	MMM	TTT	AAA	AAA	CCC	PPP	PPP
MMM	MMM	TTT	AAA	AAA	CCC	PPP	PPP
MMM	MMM	TTT	AAA	AAA	CCC	PPP	PPP
MMM	MMM	TTT	AAA	AAA	CCC	PPPPPPPPPPPP	
MMM	MMM	TTT	AAA	AAA	CCC	PPPPPPPPPPPP	
MMM	MMM	TTT	AAA	AAA	CCC	PPPPPPPPPPPP	
MMM	MMM	TTT	AAAAAAAAAAAAAAAA	AAAAAAAAAAAAAAAA	CCC	PPP	
MMM	MMM	TTT	AAAAAAAAAAAAAAAA	AAAAAAAAAAAAAAAA	CCC	PPP	
MMM	MMM	TTT	AAAAAAAAAAAAAAAA	AAAAAAAAAAAAAAAA	CCC	PPP	
MMM	MMM	TTT	AAA	AAA	CCC	PPP	
MMM	MMM	TTT	AAA	AAA	CCC	PPP	
MMM	MMM	TTT	AAA	AAA	CCC	PPP	
MMM	MMM	TTT	AAA	AAA	CCC	PPP	
MMM	MMM	TTT	AAA	AAA	CCC	PPP	
MMM	MMM	TTT	AAA	AAA	CCC	PPP	
MMM	MMM	TTT	AAA	AAA	CCCCCCCCCCCC	PPP	
MMM	MMM	TTT	AAA	AAA	CCCCCCCCCCCC	PPP	
MMM	MMM	TTT	AAA	AAA	CCCCCCCCCCCC	PPP	

```

CCCCCCCC HH      HH   KK          KK      AAAAAA CCCCCCCC CCCCCCCC
CCCCCCCC HH      HH   KK          KK      AAAAAA CCCCCCCC CCCCCCCC
CC        HH      HH   KK          KK      AA         AA CC        CC
CC        HH      HH   KK          KK      AA         AA CC        CC
CC        HH      HH   KK          KK      AA         AA CC        CC
CC        HH      HH   KK          KK      AA         AA CC        CC
CC        HHHHHHHHHH KKKKKK      AA         AA CC        CC
CC        HHHHHHHHHH KKKKKK      AA         AA CC        CC
CC        HH      HH   KK          KK      AAAAAAAAAA CC        CC
CC        HH      HH   KK          KK      AAAAAAAAAA CC        CC
CC        HH      HH   KK          KK      AA         AA CC        CC
CC        HH      HH   KK          KK      AA         AA CC        CC
CC        HH      HH   KK          KK      AA         AA CC        CC
CCCCCCCC HH      HH   KK          KK      AA         AA CCCCCCCC CCCCCCCC
CCCCCCCC HH      HH   KK          KK      AA         AA CCCCCCCC CCCCCCCC

```

```

LL          IIIII
LL          IIIII
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LL          II
LLLLLLLLLLL IIIII
LLLLLLLLLLL IIIII

SSSSSSSSS
SSSSSSSSS
SS
SS
SS
SS
SSSSSS
SSSSSS
SS
SS
SS
SS
SSSSSSSSS
SSSSSSSSS

```

```
1 0001 0 MODULE CHKACC (LANGUAGE (BLISS32) ,
2 0002 0 IDENT = 'V04-000' ,
3 0003 0 ) =
4 0004 1 BEGIN
5 0005 1
6 0006 1 *****
7 0007 1 *
8 0008 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
9 0009 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
10 0010 1 * ALL RIGHTS RESERVED.
11 0011 1 *
12 0012 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
13 0013 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
14 0014 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
15 0015 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
16 0016 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
17 0017 1 * TRANSFERRED.
18 0018 1 *
19 0019 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
20 0020 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
21 0021 1 * CORPORATION.
22 0022 1 *
23 0023 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
24 0024 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
25 0025 1 *
26 0026 1 *
27 0027 1 *****
28 0028 1
29 0029 1 ++
30 0030 1
31 0031 1 FACILITY: MTAACP
32 0032 1
33 0033 1 ABSTRACT:
34 0034 1 This routine checks that the access requested is allowed on the
35 0035 1 volume set.
36 0036 1
37 0037 1 ENVIRONMENT:
38 0038 1
39 0039 1 Starlet operating system, including privileged system services
40 0040 1 and internal exec routines.
41 0041 1
42 0042 1 --
43 0043 1
44 0044 1
45 0045 1
46 0046 1 AUTHOR: D. H. Gillespie, CREATION DATE: 17-MAY-77 09:30
47 0047 1
48 0048 1 MODIFIED BY:
49 0049 1
50 0050 1 V03-006 LMP0246 L. Mark Pilant, 2-May-1984 10:10
51 0051 1 Correct a bug introduced by LMP0221. The UCB and PCB addresses
52 0052 1 were swapped in the EXE$CHKxxxACCES routine calls.
53 0053 1
54 0054 1 V03-005 MMD0286 Meg Dumont, 10-Apr-1984 14:14
55 0055 1 Fix to the $MTACCESS returns where ACCESS might get
56 0056 1 over written with a success code before all the
57 0057 1 error conditions were checked. Fix to set the VCB
```

```
58      0058 1      FIL_ACCESS bit in KERNEL mode.
59      0059 1
60      0060 1      V03-004 LMP0221      L. Mark Pilant,      28-Mar-1984 10:21
61      0061 1      Change UCB$$_OWNUI to ORB$$_OWNER and UCB$$_VPROT to
62      0062 1      ORB$$_PROT.
63      0063 1
64      0064 1      V03-003 MMD0274      Meg Dumont,      23-Mar-1984 9:48
65      0065 1      Change the processing of the accessibility character fields
66      0066 1      in the HDR1 label to call the installation
67      0067 1      specific accessibility routine. The return from this
68      0068 1      routine determines the users access to the file. This
69      0069 1      module has also been changed to support the bit
70      0070 1      VCB$$_FIL_ACCESS which is set to determine whether
71      0071 1      VMS protection is valid for the file.
72      0072 1
73      0073 1      V03-002 MMD0239      Meg Dumont,      21-Feb-1984 10:11
74      0074 1      Change calls to EXE$CHKxxxACCES to kernel mode calls.
75      0075 1
76      0076 1      V03-001 MMD0150      Meg Dumont,      26-Apr-1983 8:51
77      0077 1      Change reference to 80 to the symbol ANSI_LBLSZ. Change
78      0078 1      reference to 240 to the symbol SCRATCH_OFFSET.
79      0079 1
80      0080 1      V02-007 DMW00032      David Micahel Walp      18-Aug-1981
81      0081 1      Looked at MVL Override Bit when override option is used
82      0082 1
83      0083 1      V02-006 REFORMAT      Maria del C. Nasr      30-Jun-1980
84      0084 1
85      0085 1      A0005 MCN0003      Maria del C. Nasr      15-Oct-1979 9:29
86      0086 1      Add HDR3 processing
87      0087 1
88      0088 1      A0004 MCN0001      Maria del C. Nasr      13-Sep-79 11:05
89      0089 1      Corrected bug in "create if" function
90      0090 1
91      0091 1      **
92      0092 1
93      0093 1      LIBRARY 'SYS$LIBRARY:LIB.L32';
94      0094 1
95      0095 1      REQUIRE 'SRC$:MTADEF.B32';
96      0479 1
97      0480 1      FORWARD ROUTINE
98      0481 1      CHECK_ACCESS      : COMMON_CALL NOVALUE,      ! check access
99      0482 1      CHECK_FILE_ACC      : COMMON_CALL NOVALUE,      ! check access to file
100     0483 1      CHECK_WRITE_ACCESS : COMMON_CALL,      ! check users' write access
101     0484 1      CHECK_READ_ACCESS  : COMMON_CALL,      ! check users' write access
102     0485 1      SET_FILE_ACCESS    : COMMON_CALL NOVALUE,      ! Set VCB file access
103     0486 1      RECALC_ST_REC      : COMMON_CALL NOVALUE;      ! recalculate start record
104     0487 1
105     0488 1      LINKAGE
106     0489 1      CHECK_PROT      = JSB (REGISTER = 4, REGISTER = 5) :
107     0490 1      NOPRESERVE (1, 2, 3);
108     0491 1
109     0492 1      EXTERNAL ROUTINE
110     0493 1      EXE$CHKWRTACCES : ADDRESSING_MODE (ABSOLUTE) CHECK_PROT,
111     0494 1      EXE$CHKRDACCES  : ADDRESSING_MODE (ABSOLUTE) CHECK_PROT,
112     0495 1      GET_RECORD;      ! get current record tape is reading
113     0496 1
114     0497 1      EXTERNAL
```

CHKACC  
V04-000

L 14  
16-Sep-1984 02:09:19 VAX-11 Bliss-32 V4.0-742  
14-Sep-1984 12:46:34 [MTAACP.SRC]CHKACC.B32;1

Page 3  
(1)

:	115	0498	1	CURRENT_UCB	:	REF BBLOCK,	!	address of current ucb
:	116	0499	1	IO_PACKET	:	REF BBLOCK,	!	address of current io request packet
:	117	0500	1	USER_STATUS	:	WORD;	!	address of status to return to user
:	118	0501	1					

```
120 0502 1 GLOBAL ROUTINE CHECK_ACCESS (ACCESS_TYPE) : COMMON_CALL NOVALUE =
121 0503 1
122 0504 1 ++
123 0505 1
124 0506 1 FUNCTIONAL DESCRIPTION:
125 0507 1 This routine checks that the access requested is allowed on the
126 0508 1 volume set.
127 0509 1
128 0510 1 CALLING SEQUENCE:
129 0511 1 CHECK_ACCESS(ARG1)
130 0512 1
131 0513 1 INPUT PARAMETERS:
132 0514 1 ARG1 - access requested (0=read,1=write)
133 0515 1
134 0516 1 IMPLICIT INPUTS:
135 0517 1 IO_PACKET - address of current i/o packet
136 0518 1 CURRENT_UCB - address of current ucb
137 0519 1
138 0520 1 OUTPUT PARAMETERS:
139 0521 1 None
140 0522 1
141 0523 1 IMPLICIT OUTPUTS:
142 0524 1 None
143 0525 1
144 0526 1 ROUTINE VALUE:
145 0527 1 None
146 0528 1
147 0529 1 SIDE EFFECTS:
148 0530 1 None
149 0531 1 USER ERROR:
150 0532 1 SSS_WRITLCK - software write lock
151 0533 1
152 0534 1 --
153 0535 1
154 0536 2 BEGIN
155 0537 2
156 0538 2 EXTERNAL REGISTER
157 0539 2 COMMON_REG;
158 0540 2
159 0541 2 LOCAL
160 0542 2 STATUS; ! io status
161 0543 2
162 0544 2 ! If file is software write locked and the user requests write privileges,
163 0545 2 ! deny privilege
164 0546 2
165 0547 2 IF .ACCESS_TYPE
166 0548 2 AND
167 0549 2 .BBLOCK[CURRENT_UCB[UCB$L_DEVCHAR], DEV$V_SWL]
168 0550 2 THEN
169 0551 2 ERR_EXIT(SSS_WRITLCK);
170 0552 2
171 0553 2 ! If the VCB$V_FIL_ACCESS is set then the user has complete
172 0554 2 ! access to this file, regardless of how the VMS protection is
173 0555 2 ! set. Else check the users read and write access to the file.
174 0556 2
175 0557 2 IF NOT .CURRENT_VCB[VCB$V_FIL_ACCESS]
176 0558 2 THEN
```

CHKACC  
V04-000

N 14  
16-Sep-1984 02:09:19  
14-Sep-1984 12:46:34

VAX-11 Bliss-32 V4.0-742  
[MTAACP.SRC]CHKACC.B32;1

Page 5  
(2)

```
: 177      0559  3      BEGIN
: 178      0560  3      IF .ACCESS_TYPE
: 179      0561  3      THEN
: 180      0562  3      STATUS = KERNEL_CALL (CHECK_WRITE_ACCESS)
: 181      0563  3      ELSE
: 182      0564  3      STATUS = KERNEL_CALL (CHECK_READ_ACCESS);
: 183      0565  3
: 184      0566  3      IF NOT .STATUS
: 185      0567  3      THEN
: 186      0568  3      BEGIN
: 187      0569  3      USER_STATUS = .STATUS<0, 16>;
: 188      0570  3      ERR_EXIT();
: 189      0571  3      END;
: 190      0572  3      END;
: 191      0573  2
: 192      0574  1      END;
```

! end of routine

```
.TITLE  CHKACC
.IDENT  \V04-000\

.EXTRN  EX$CHKWRTACCES
.EXTRN  EX$CHKRDACCES, GET_RECORD
.EXTRN  CURRENT_UCB, IO_PACKET
.EXTRN  USER_STATUS, SYS$CMKRNL
```

.PSECT \$CODE\$,NOWRT,2

```
.ENTRY  CHECK_ACCESS, Save nothing
BLBC    ACCESS_TYPE, 1$
MOVL    CURRENT_UCB, R0
BBC     #1, 59(R0), 1$
CHMU    #604
BBS     #6, 45(CURRENT_VCB), 4$
BLBC    ACCESS_TYPE, 2$
CLRL    -(SP)
PUSHL   SP
PUSHAB  CHECK_WRITE_ACCESS
BRB     3$
CLRL    -(SP)
PUSHL   SP
PUSHAB  CHECK_READ_ACCESS
CALLS   #3, @SYS$CMKRNL
BLBS    STATUS, 4$
MOVW    STATUS, USER_STATUS
CHMU    #0
RET
```

```
: 0502
: 0547
: 0549
: 0551
: 0557
: 0560
: 0562
: 0564
: 0566
: 0569
: 0570
: 0574
```

; Routine Size: 65 bytes, Routine Base: \$CODE\$ + 0000

; 193 0575 1

```
195 0576 1 ROUTINE CHECK_WRITE_ACCESS : COMMON_CALL =
196 0577 1
197 0578 1 ++
198 0579 1
199 0580 1 FUNCTIONAL DESCRIPTION:
200 0581 1 This routine calls the system routine to check users write access.
201 0582 1
202 0583 1 CALLING SEQUENCE:
203 0584 1 CHECK_WRITE_ACCESS (), called in kernel mode
204 0585 1
205 0586 1 INPUT PARAMETERS:
206 0587 1 none
207 0588 1
208 0589 1 IMPLICIT INPUTS:
209 0590 1 CURRENT_UCB - address of tapes ucb
210 0591 1 IO_PACKET - address of current io request
211 0592 1
212 0593 1 OUTPUT PARAMETERS:
213 0594 1 None
214 0595 1
215 0596 1 IMPLICIT OUTPUTS:
216 0597 1 None
217 0598 1
218 0599 1 ROUTINE VALUE:
219 0600 1 STATUS from call
220 0601 1
221 0602 1 SIDE EFFECTS:
222 0603 1
223 0604 1 USER ERROR:
224 0605 1
225 0606 1 --
226 0607 1
227 0608 2 BEGIN
228 0609 2
229 0610 2 EXTERNAL REGISTER
230 0611 2 COMMON_REG;
231 0612 2
232 0613 2 LOCAL
233 0614 2 PCB : REF BBLOCK; ! address of user process control block
234 0615 2
235 0616 2 EXTERNAL
236 0617 2 SCH$GL_PCBVEC : REF VECTOR ADDRESSING_MODE (ABSOLUTE);
237 0618 2 ! system PCB vector
238 0619 2
239 0620 2 PCB = .SCH$GL_PCBVEC[(IO_PACKET[IRPSL_PID])<0,16>];
240 0621 2
241 0622 2 RETURN EX$CHKWRTACCES(.PCB, .CURRENT_UCB);
242 0623 1 END;
```

.EXTRN SCH\$GL\_PCBVEC

07FC 00000 CHECK\_WRITE\_ACCESS:

```
51 00000000G 9F D0 00002 .WORD Save R2,R3,R4,R5,R6,R7,R8,R9,R10
50 0000G CF D0 00009 MOVL @#SCH$GL_PCBVEC, R1
MOVL IO_PACKET, R0
```

: 0576  
: 0620  
:

```
ADDL2    #12, R0
MOVZWL   (R0), R0
MOVL     (R1)(R0), PC
MOVL     CURRENT_UCB, R5
JSB      @#EXESCHKWRTACCS
RET
```

0622  
0623

```
; Routine Size: 36 bytes,    Routine Base: $CODE$ + 0041
```

```
244 0624 1 ROUTINE CHECK_READ_ACCESS : COMMON_CALL =
245 0625 1
246 0626 1 ++
247 0627 1
248 0628 1 FUNCTIONAL DESCRIPTION:
249 0629 1 This routine returns users' read access to the file.
250 0630 1
251 0631 1 CALLING SEQUENCE:
252 0632 1 CHECK_READ_ACCESS(), called in kernel mode
253 0633 1
254 0634 1 INPUT PARAMETERS:
255 0635 1 none
256 0636 1
257 0637 1 IMPLICIT INPUTS:
258 0638 1 CURRENT_UCB - address of tapes ucb
259 0639 1 IO_PACKET - address of current io request
260 0640 1
261 0641 1 OUTPUT PARAMETERS:
262 0642 1 None
263 0643 1
264 0644 1 IMPLICIT OUTPUTS:
265 0645 1 None
266 0646 1
267 0647 1 ROUTINE VALUE:
268 0648 1 STATUS from call
269 0649 1
270 0650 1 SIDE EFFECTS:
271 0651 1
272 0652 1 USER ERROR:
273 0653 1
274 0654 1 --
275 0655 1
276 0656 2 BEGIN
277 0657 2
278 0658 2 EXTERNAL REGISTER
279 0659 2 COMMON_REG;
280 0660 2
281 0661 2 LOCAL
282 0662 2 PCB : REF BBLOCK; ! address of user process control block
283 0663 2
284 0664 2 EXTERNAL
285 0665 2 SCH$GL_PCBVEC : REF VECTOR ADDRESSING_MODE (ABSOLUTE);
286 0666 2 ! system PCB vector
287 0667 2
288 0668 2 PCB = .SCH$GL_PCBVEC[.(IO_PACKET[IRPSL_PID])<0,16>];
289 0669 2
290 0670 2 RETURN EX$CHKRDACCES(.PCB, .CURRENT_UCB);
291 0671 1 END;
```

07FC 00000 CHECK\_READ\_ACCESS:

```
51 00000000G 9F D0 00002 .WORD Save R2,R3,R4,R5,R6,R7,R8,R9,R10
50 0000G CF D0 00009 MOVL @#SCH$GL_PCBVEC, R1
MOVL IO_PACKET, R0
```

: 0624  
: 0668  
:

CHKACC  
V04-000

E 15  
16-Sep-1984 02:09:19  
14-Sep-1984 12:46:34

VAX-11 Bliss-32 V4.0-742  
[MTAACP.SRC]CHKACC.B32;1

Page 9  
(4)

50		0C	C0	0000E
50		60	3C	00011
54		6140	D0	00014
55	0000G	CF	D0	00018
	00000000G	9F	16	0001D
			04	00023

ADDL2	#12, R0
MOVZWL	(R0), R0
MOVL	(R1)(R0), PCB
MOVL	CURRENT UCB, R5
JSB	@#EXESCHKRDACCES
RET	

0670  
0671

; Routine Size: 36 bytes, Routine Base: \$CODE\$ + 0065

: 292	0672	1
: 293	0673	1

```
295 0674 1 GLOBAL ROUTINE CHECK_FILE_ACC (ACCESS_CALL) : COMMON_CALL NOVALUE =
296 0675 1
297 0676 1 ++
298 0677 1
299 0678 1 FUNCTIONAL DESCRIPTION:
300 0679 1 This routine checks access to the file. If accessibility code is
301 0680 1 not blank and not overridden then access is denied. If writing to the file the file
302 0681 1 must be expired.
303 0682 1
304 0683 1 CALLING SEQUENCE:
305 0684 1 CHECK_FILE_ACC(ARG1)
306 0685 1
307 0686 1 INPUT PARAMETERS:
308 0687 1 0 - If being called from MTA_CREATE
309 0688 1 1 - If called from MTA_ACCESS
310 0689 1
311 0690 1 IMPLICIT INPUTS:
312 0691 1 LOCAL_FIB - copy of user's file information block
313 0692 1 CURRENT_VCB - address of current control block
314 0693 1
315 0694 1 OUTPUT PARAMETERS:
316 0695 1 None
317 0696 1
318 0697 1 IMPLICIT OUTPUTS:
319 0698 1 None
320 0699 1
321 0700 1 ROUTINE VALUE:
322 0701 1 None
323 0702 1
324 0703 1 SIDE EFFECTS:
325 0704 1 if append, tape is positioned to end of data
326 0705 1
327 0706 1 USER ERROR:
328 0707 1 $$$_FILACCERR - file access byte non-blank
329 0708 1
330 0709 1 --
331 0710 1
332 0711 2 BEGIN
333 0712 2
334 0713 2 EXTERNAL REGISTER
335 0714 2 COMMON_REG;
336 0715 2
337 0716 2 EXTERNAL ROUTINE
338 0717 2 EXPIRED : COMMON_CALL, ! check that file has expired
339 0718 2 LIB$CVT_DTB : ADDRESSING_MODE (ABSOLUTE),
340 0719 2
341 0720 2 SPACE_EOF : COMMON_CALL, ! convert decimal to binary
342 0721 2 SPACE_TM : COMMON_CALL, ! space to trailers
343 0722 2
344 0723 2 READ_BLOCK : COMMON_CALL; ! space given number of
345 0724 2
346 0725 2 LOCAL
347 0726 2 ACCESS, ! users' access to the file
348 0727 2 BLOCK_COUNT, ! block count of file to
349 0728 2
350 0729 2 CURRENT_RECORD, ! appended to
351 0730 2 FIB : REF BBLOCK, ! record tape drive is reading
! address of local fib
```

```
352 0731 2 LABELADDR : REF BBLOCK, ! address of label
353 0732 2 STATUS,
354 0733 2 TM, ! number of tm's
355 0734 2 ORB : REF BBLOCK, ! ORB address
356 0735 2 MVL : REF BBLOCK,
357 0736 2 MVL_ENTRY : REF BBLOCKVECTOR [,MVL$K_LENGTH];
358 0737 2 ! pointer to the MVL_ENTRY
359 0738 2
360 0739 2 EXTERNAL
361 0740 2 HDR1 : REF BBLOCK, ! address of HDR1(EOF1) label
362 0741 2 LOCAL_FIB : BBLOCK; ! copy of user's fib
363 0742 2
364 0743 2 ! setup pointer to fib
365 0744 2
366 0745 2 FIB = LOCAL_FIB;
367 0746 2
368 0747 2 ! get a handle on the MVL entry
369 0748 2
370 0749 2 MVL = .CURRENT_VCB[VCB$MVL];
371 0750 2 MVL_ENTRY = (.CURRENT_VCB[VCB$MVL]) + MVL$K_FIXLEN;
372 0751 2
373 0752 2 ! Call the accessibility system service to check the accessibility char
374 0753 2 on the HDR1 label.
375 0754 2 ! First keep the record that the UCB is reading. The accessibility
376 0755 2 routine can not move the tape from under us! Thus we will compare
377 0756 2 this to the field after the call and if the tape was moved we punt
378 0757 2 ! the operation.
379 0758 2
380 0759 2 ORB = .CURRENT_UCB[UCB$ORB];
381 0760 2 CURRENT_RECORD = KERNEL CALL(GET_RECORD, .CURRENT_UCB);
382 0761 2 ACCESS = $MTACCESS(LBLNAM = HDR1,
383 0762 2 UIC = .ORB[ORB$OWNER],
384 0763 2 STD_VERSION = .MVL[MVL$STDVER],
385 0764 2 ACCESS_CHAR = 0,
386 0765 2 ACCESS_SPEC = MTASK_NOCHAR,
387 0766 2 TYPE = MTASK_INHDR1);
388 0767 2
389 0768 2 STATUS = KERNEL CALL(GET_RECORD, .CURRENT_UCB);
390 0769 2 IF .CURRENT_RECORD NEQ .STATUS
391 0770 2 THEN ERR_EXIT(SS$_TAPEPOSLOST);
392 0771 2
393 0772 2 IF .ACCESS EQL SS$_FILACCERR
394 0773 2 THEN
395 0774 2 BEGIN
396 0775 2 IF NOT ( .CURRENT_VCB[VCB$V_OVRACC]
397 0776 2 AND .MVL_ENTRY[ (.CURRENT_VCB[VCB$W_RVN]), MVL$V_OVERRIDE ])
398 0777 2 THEN ERR_EXIT(SS$_FILACCERR);
399 0778 2 ACCESS = SS$_NORMAL;
400 0779 2 END;
401 0780 2
402 0781 2 IF .ACCESS EQL SS$_NOVOLACC OR .ACCESS EQL SS$_NOFILACC
403 0782 2 THEN ERR_EXIT(.ACCESS);
404 0783 2
405 0784 2 IF NOT .ACCESS THEN KERNEL_CALL(SET_FILE_ACCESS);
406 0785 2
407 0786 2 ! now treat append case uniquely
408 0787 2
```

```

409      0788 3
410      0789
411      0790
412      0791
413      0792
414      0793
415      0794
416      0795
417      0796
418      0797
419      0798
420      0799
421      0800
422      0801
423      0802
424      0803
425      0804
426      0805
427      0806
428      0807 4
429      0808 4
430      0809 4
431      0810 4
432      0811 3
433      0812
434      0813
435      0814
436      0815
437      0816
438      0817
439      0818
440      0819
441      0820
442      0821
443      0822
444      0823
445      0824
446      0825
447      0826
448      0827
449      0828
450      0829
451      0830
452      0831
453      0832
454      0833
455      0834
456      0835
457      0836 1

IF ( NOT .FIB[FIB$V_UPDATE])
AND
.FIB[FIB$V_WRITE]
AND
(.ACCESS_CALL)
THEN
BEGIN
SPACE_EOF();
! append case
! left at absolute end of file
IF NOT LIB$CVT_DTB(E01$S_BLOCKCNT, HDR1[E01$T_BLOCKCNT], BLOCK_COUNT)
THEN
ERR_EXIT(SS$_BLOCKCNTERR);
! read header of next file
LABELADDR = .HDR1 + SCRATCH_OFFSET; ! read into scratch area
IF NOT READ_BLOCK(.LABELADDR, ANSI_LBLSZ)
THEN
BEGIN
SPACE_TM(-3); ! at logical end of volume set
! double tape mark is logical end of tape
KERNEC_CALL(RECALC_ST_REC, .BLOCK_COUNT);
RETURN;
END;
IF .LABELADDR[HD1$L_HD1LID] NEQ 'HDR1'
THEN ERR_EXIT(SS$_TAPEPOSLOST);
! going to overlay file
IF NOT EXPIRED(LABELADDR[HD1$T_EXPIREDT])
THEN ERR_EXIT(SS$_FILNOTEXP);
SPACE_TM(-2);
KERNEC_CALL(RECALC_ST_REC, .BLOCK_COUNT);
RETURN;
END;
! end of append case
! if about to write current file check expiration
IF .FIB[FIB$V_WRITE]
THEN
IF NOT EXPIRED(HDR1[HD1$T_EXPIREDT])
THEN
ERR_EXIT(SS$_FILNOTEXP);
END;
! end of routine
```

```

.EXTRN EXPIRED, LIB$CVT_DTB
.EXTRN SPACE_EOF, SPACE_TM
.EXTRN READ_BLOCK, HDR1
.EXTRN LOCAL_FIB, SYSSMTACCESS
```

			01FC	00000	.ENTRY	CHECK_FILE_ACC, Save R2,R3,R4,R5,R6,R7,R8	0674	
	58	0000G	CF	9E	00002	MOVAB	HDR1, R8	
	57	00000000G	9F	9E	00007	MOVAB	@#SYSSCMKRNL, R7	
	5E		04	C2	0000E	SUBL2	#4, SP	
	55	0000G	CF	9E	00011	MOVAB	LOCAL FIB, FIB	0745
53	52	34	AB	D0	00016	MOVL	52(CURRENT_VCB), MVL	0749
	51		24	C1	0001A	ADDL3	#36, 52(CURRENT_VCB), MVL_ENTRY	0750
	50	0000G	CF	D0	0001F	MOVL	CURRENT_UCB, R0	0759
	54	1C	A0	D0	00024	MOVL	28(R0), -ORB	
			50	DD	00028	PUSHL	R0	0760
			01	DD	0002A	PUSHL	#1	
			5E	DD	0002C	PUSHL	SP	
		0000G	CF	9F	0002E	PUSHAB	GET_RECORD	
	67		04	FB	00032	CALLS	#4, -SYSSCMKRNL	
	56		50	D0	00035	MOVL	R0, CURRENT_RECORD	
			01	DD	00038	PUSHL	#1	0766
	7E	22	7E	7C	0003A	CLRG	-(SP)	
			A2	9A	0003C	MOVZBL	34(MVL), -(SP)	
			64	DD	00040	PUSHL	(ORB)	
			68	DD	00042	PUSHL	HDR1	
	00000000G	00	06	FB	00044	CALLS	#6, SYSSMTACCESS	
		52	50	D0	0004B	MOVL	R0, ACCESS	
		0000G	CF	DD	0004E	PUSHL	CURRENT_UCB	0768
			01	DD	00052	PUSHL	#1	
			5E	DD	00054	PUSHL	SP	
		0000G	CF	9F	00056	PUSHAB	GET_RECORD	
	67		04	FB	0005A	CALLS	#4, -SYSSCMKRNL	
	50		56	D1	0005D	CMPL	CURRENT_RECORD, STATUS	0769
			04	13	00060	BEQL	1\$	
	0000009C	8F	8F	BF	00062	CHMU	#548	0770
			52	D1	00066	CMPL	ACCESS, #156	0772
			18	12	0006D	BNEQ	4\$	
0C	2C	AB	01	E1	0006F	BBC	#1, 44(CURRENT_VCB), 2\$	0775
		50	0E	AB	3C	MOVZWL	14(CURRENT_VCB), R0	0776
		07	A340	7F	00078	PUSHAQ	7(MVL_ENTRY)[R0]	
04		9E	02	E0	0007C	BBS	#2, @TSP)+, 3\$	
		009C	8F	BF	00080	CHMU	#156	0777
			01	D0	00084	MOVL	#1, ACCESS	0778
	000022A4	8F	52	D1	00087	CMPL	ACCESS, #8868	0781
			09	13	0008E	BEQL	5\$	
	000022AC	8F	52	D1	00090	CMPL	ACCESS, #8876	
			02	12	00097	BNEQ	6\$	
			52	BF	00099	CHMU	ACCESS	0782
		0B	52	E8	0009B	BLBS	ACCESS, 7\$	0784
			7E	D4	0009E	CLRL	-(SP)	
			5E	DD	000A0	PUSHL	SP	
		0000V	CF	9F	000A2	PUSHAB	SET_FILE_ACCESS	
	67		03	FB	000A6	CALLS	#3, -SYSSCMKRNL	
70	65		06	E0	000A9	BBS	#6, (FIB), 13\$	0788
	6C	01	A5	E9	000AD	BLBC	1(FIB), 13\$	0790
	68	04	AC	E9	000B1	BLBC	ACCESS_CALL, 13\$	0792
		0000G	00	FB	000B5	CALLS	#0, SPACE_EOF	0795
			5E	DD	000BA	PUSHL	SP	0797
	7E	68	36	C1	000BC	ADDL3	#54, HDR1, -(SP)	
			06	DD	000C0	PUSHL	#6	
	00000000G	9F	03	FB	000C2	CALLS	#3, @#LIB\$CVT_DTB	
		04	50	E8	000C9	BLBS	R0, 8\$	

53	68	0940	8F	BF	000CC	CHMU	#2368	:	0799
	7E	00000140	8F	C1	000D0	ADDL3	#320, HDR1, LABELADDR	:	0803
		50	8F	9A	000D8	MOVZBL	#80, -(SP)	:	0805
			53	DD	000DC	PUSHL	LABELADDR	:	
	0000G		02	FB	000DE	CALLS	#2, READ_BLOCK	:	
	05		50	E8	000E3	BLBS	R0, 9\$	:	
	7E		03	CE	000E6	MNEGL	#3, -(SP)	:	0808
			1F	11	000E9	BRB	12\$	:	
31524448	8F		63	D1	000EB	9\$: CMPL	(LABELADDR), #827475016	:	0813
			04	13	000F2	BEQL	10\$	:	
		0224	8F	BF	000F4	CHMU	#548	:	0814
		2F	A3	9F	000F8	10\$: PUSHAB	47(LABELADDR)	:	0818
	0000G		01	FB	000FB	CALLS	#1, EXPIRED	:	
	04		50	E8	00100	BLBS	R0, 11\$	:	
		00B4	8F	BF	00103	CHMU	#180	:	0819
	7E		02	CE	00107	11\$: MNEGL	#2, -(SP)	:	0821
	0000G		01	FB	0010A	12\$: CALLS	#1, SPACE_TM	:	
	CF		6E	DD	0010F	PUSHL	BLOCK_COUNT	:	0822
			01	DD	00111	PUSHL	#1	:	
			5E	DD	00113	PUSHL	SP	:	
		0000V	CF	9F	00115	PUSHAB	RECALC_ST_REC	:	
	67		04	FB	00119	CALLS	#4, SYSSCMKRNL	:	
			04	0011C	RET			:	0794
	10	01	A5	E9	0011D	13\$: BLBC	1(FIB), 14\$	:	0829
7E	68		2F	C1	00121	ADDL3	#47, HDR1, -(SP)	:	0832
	0000G		01	FB	00125	CALLS	#1, EXPIRED	:	
	CF		50	E8	0012A	BLBS	R0, 14\$	:	
	04		8F	BF	0012D	CHMU	#180	:	0834
		00B4	04	00131	14\$: RET			:	0836

; Routine Size: 306 bytes, Routine Base: \$CODE\$ + 0089

; 458 0837 1

```
: 460      0838 1 ROUTINE SET_FILE_ACCESS : COMMON_CALL NOVALUE =
: 461      0839 1
: 462      0840 1 ++
: 463      0841 1
: 464      0842 1 FUNCTIONAL DESCRIPTION:
: 465      0843 1     This routine updates the VCB file access bit to say that the user
: 466      0844 1     has complete access to the file.
: 467      0845 1
: 468      0846 1 CALLING SEQUENCE:
: 469      0847 1     SET_FILE_ACCESS called in kernel mode
: 470      0848 1
: 471      0849 1 INPUT PARAMETERS:
: 472      0850 1     none
: 473      0851 1
: 474      0852 1 IMPLICIT INPUTS:
: 475      0853 1     CURRENT_VCB - address of current volume control block
: 476      0854 1
: 477      0855 1 OUTPUT PARAMETERS:
: 478      0856 1     None
: 479      0857 1
: 480      0858 1 IMPLICIT OUTPUTS:
: 481      0859 1     VCB FIL_ACCESS bit is set.
: 482      0860 1
: 483      0861 1 ROUTINE VALUE:
: 484      0862 1     None
: 485      0863 1
: 486      0864 1 SIDE EFFECTS:
: 487      0865 1     None
: 488      0866 1
: 489      0867 1 USER ERRORS:
: 490      0868 1     None
: 491      0869 1
: 492      0870 1 --
: 493      0871 1
: 494      0872 2 BEGIN
: 495      0873 2
: 496      0874 2 EXTERNAL REGISTER
: 497      0875 2     COMMON_REG;
: 498      0876 2
: 499      0877 2     CURRENT_VCB[VCB$V_FIL_ACCESS] = 1;
: 500      0878 1 END;                                     ! end of routine
```

```
                                0000 00000 SET_FILE_ACCESS:
                                .WORD      Save nothing
                                BISB2      #64, 45(CURRENT_VCB)
                                RET
                                : 0838
                                : 0877
                                : 0878
```

; Routine Size: 8 bytes,      Routine Base: \$CODE\$ + 01BB

```
: 501      0879 1
: 502      0880 1
```

```
: 504      0881 1 ROUTINE RECALC_ST_REC (BLOCK_COUNT) : COMMON_CALL NOVALUE =
: 505      0882 1
: 506      0883 1 ++
: 507      0884 1
: 508      0885 1 FUNCTIONAL DESCRIPTION:
: 509      0886 1     This routine updates the start record count to include those
: 510      0887 1     records in the file that were previously written.
: 511      0888 1
: 512      0889 1 CALLING SEQUENCE:
: 513      0890 1     RECALC_ST_REC(ARG1), called in kernel mode
: 514      0891 1
: 515      0892 1 INPUT PARAMETERS:
: 516      0893 1     ARG1 - number of blocks previously written
: 517      0894 1
: 518      0895 1 IMPLICIT INPUTS:
: 519      0896 1     CURRENT_VCB - address of current volume control block
: 520      0897 1
: 521      0898 1 OUTPUT PARAMETERS:
: 522      0899 1     None
: 523      0900 1
: 524      0901 1 IMPLICIT OUTPUTS:
: 525      0902 1     Start record number updated to reflect previously written records
: 526      0903 1
: 527      0904 1 ROUTINE VALUE:
: 528      0905 1     None
: 529      0906 1
: 530      0907 1 SIDE EFFECTS:
: 531      0908 1     None
: 532      0909 1
: 533      0910 1 USER ERRORS:
: 534      0911 1     None
: 535      0912 1
: 536      0913 1 --
: 537      0914 1
: 538      0915 2 BEGIN
: 539      0916 2
: 540      0917 2 EXTERNAL REGISTER
: 541      0918 2     COMMON_REG;
: 542      0919 2
: 543      0920 2 CURRENT_VCB[VCB$$_ST_RECORD] = .CURRENT_VCB[VCB$$_ST_RECORD] -
: 544      0921 2     .BLOCK_COUNT;
: 545      0922 1 END;                                ! end of routine
```

```
                                0000 00000 RECALC_ST_REC:
                                30  AB      04  AC  C2 00002  .WORD  Save nothing
                                04 00007  SUBL2  BLOCK_COUNT, 48(CURRENT_VCB)
                                RET
: Routine Size: 8 bytes,      Routine Base: $CODE$ + 01C3
: 546      0923 1 END
: 547      0924 1
```

: 0881  
: 0921  
: 0922

CHKACC  
V04-000

: 548

0925 0 ELUDOM

M 15  
16-Sep-1984 02:09:19  
14-Sep-1984 12:46:34

VAX-11 Bliss-32 V4.0-742  
[MTAACP.SRC]CHKACC.B32;1

Page 17  
(7)

PSECT SUMMARY

Name	Bytes	Attributes
\$CODE\$	459	NOVEC,NOWRT, RD , EXE,NOSHR, LCL, REL, CON,NOPI,ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[SYSLIB]LIB.L32;1	18619	37	0	1000	00:01.8

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LISS:CHKACC/OBJ=OBJ\$:CHKACC MSRC\$:CHKACC/UPDATE=(ENH\$:CHKACC)

: Size: 459 code + 0 data bytes  
: Run Time: 00:13.5  
: Elapsed Time: 00:51.6  
: Lines/CPU Min: 4098  
: Lexemes/CPU-Min: 19364  
: Memory Used: 145 pages  
: Compilation Complete

0253

AH-BT13A-SE  
 VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION  
CONFIDENTIAL AND PROPRIETARY